Project	IEEE 802.16 Broadband Wireless Access Working Group <**http://ieee802.org/16**>	
Title	LDPC coding for OFDMA PHY	
Date Submitted	2005-1-27	
Source(s)	Brian Classon Yufei Blankenship Motorola	brian.classon@motorola.com yufei.blankenship@motorola.com
	Kyuhyuk Chung Kihyoung Cho Min-seok Oh LG Electronics	kyuhyuk@lge.com kihyoung@lge.com minoh@lge.com
	Jerry Kim Gyubum Kyung DS Park Jaeho Jeon Samsung	kimjy@samsung.com gyubum.kyung@samsung.com dspark@samsung.com jhjeon@samsung.com
	Eric Jacobsen Bo Xia Intel	eric.a.jacobsen@intel.com bo.xia@intel.com
	Dale Hocevar Anuj Batra Texas Instruments	hocevar@ti.com batra@ti.com
	Brian Johnson Alek Purkovic Nina Burns Sergey Sukobok Michael Livshitz Nortel Networks	brjohnso@nortelnetworks.com apurkovi@nortelnetworks.com nburns@nortelnetworks.com ssukobok@nortelnetworks.com livshitz@nortelnetworks.com
	Yaniv Nana Yossi Segal	yaniv.nana@runcom.co.il yossis@runcom.co.il

Eli Shasha shasha@runcom.co.il
Zion Hadad zionh@runcom.co.il
Runcom Technologies

Simon Litsyn litsyn@eng.tau.ac.il
Eran Sharon eransh@eng.tau.ac.il
Tel-Aviv University.

Walter Rausch walter.f.rausch@mail.sprint.com
Chris Seagren chris.e.seagren@mail.sprint.com
John Humbert john.j.humbert@mail.sprint.com
Sprint

Masoud Olfat masoud.olfat@nextel.com
Nextel Communications

Ron Murias RMurias@WI-LAN.com
Wi-LAN

Hebing Wu wuhebing@huawei.com
Michael Wang wangjibin@huawei.com
Huawei

Robert Xu xu.jun2@zte.com.cn
David Yuan yuan.liuqing@zte.com.cn
Li Zeng li.zeng@zte.com.cn
Liujun Hu hu.liujun@zte.com.cn
ZTE Inc.

Re:	IEEE P802.16-REVe/D5a, BRC recirc
Abstract	This contribution contains the LDPC code output from a downselection process in an informal LDPC group.
Purpose	Complete the LDPC specification text.
Notice	This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.

Release	The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE's name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE's sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that this contribution may be made public by IEEE 802.16.
Patent Policy and Procedures	The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures <http://ieee802.org/16/ipr/patents/policy.html>, including the statement "IEEE standards may include the known use of patent(s), including patent applications, provided the IEEE receives assurance from the patent holder or applicant with respect to patents essential for compliance with both mandatory and optional portions of the standard." Early disclosure to the Working Group of patent information that might be relevant to the standard is essential to reduce the possibility for delays in the development process and increase the likelihood that the draft publication will be approved for publication. Please notify the Chair <mailto:chair@wirelessman.org> as early as possible, in written or electronic form, if patented technology (or technology under patent application) might be incorporated into a draft standard being developed within the IEEE 802.16 Working Group. The Chair will disclose this notification via the IEEE 802.16 web site <http://ieee802.org/16/ipr/patents/notices>.

Overview

An informal LDPC group has been working on the goal of achieving consensus on an optional advanced LDPC code for the OFDMA PHY. Substantial text on LDPC has been previously harmonized and included within 802.16e – this contribution completes the LDPC specification text by adding code matrices fully compliant with the existing specification text. Simulation results are also provided to show that the LDPC code (selected after an extensive feature harmonization and downselection process) offers excellent performance; significantly better than convolutional codes (CC) and the same or better than the convolutional turbo codes (CTC) defined for 802.16.

The LDPC code was selected through a downselection process of eight candidates – Intel, LG, Motorola, Nokia, Nortel, Runcom, Samsung, and TI. Going into downselection, the candidates shared many desirable features through previous harmonization. For example, all proposals used structured LDPC codes, first proposed to 802.16 by Samsung. Six proposals had the same structure for the parity portion of the matrix, which prevented performance-degrading multiple weight-1 columns and allowed low-complexity differential-style-encoding (proposed by Motorola and Samsung, and similar to the encoding in the very first 802.16 LDPC proposal by Intel). Three proposals used the same scaling method of shift values for different block sizes (Motorola, Intel, and LG). Each proposal had particular features designed for a good performance / complexity tradeoff. Throughout the downselection process, codes were redesigned to reduce complexity by incorporating desired features of eliminated candidates. The downselection winner, Motorola, enhanced its proposal by making each matrix have 24 base columns for simplified decoding (desired by TI) and adding a non-intersecting row feature to its R=1/2 code for faster decoding (desired by Runcom). Nortel provided a R=3/4 code redesign with the same features and performance as the Motorola R=3/4 code, except the code was semi-regular for reduced complexity. (A semi-regular R=3/4 code was also steadfastly championed by LG throughout the downselection process.) Samsung then offered a redesigned R=2/3 code that had a compact representation similar to the Motorola code, resulting in a Motorola + Nortel + Runcom + Samsung + Intel proposal. This winning proposal passed a confirmation vote (75% threshold) within the informal LDPC group, validating the downselection process.

In a reply comment to this contribution, LG stated that they were uncomfortable with the downselection winner having an error floor at 1e-3 FER in one of the 57 different code rate / block size combinations (N=1536 R=2/3). This revision addresses that concern by revising the 2/3 matrix. All features of the revised 2/3 matrix are the same.

Further harmonization is achieved by including Huawei's parallel encoding method within the existing Method 1 optional encoding method.

In 006r3, matrices from LG and ZTE are included. The LG matrix is R=2/3, and has the Runcom non-intersecting row feature. The ZTE matrix is 3/4, and is fully irregular. All other features of the additional matrices are the same as the current matrices. All LDPC capable units must be able to process both 2/3 and both 3/4 matrices. The type of matrices used is selected via the burst profile encodings.

Features

The LDPC code has excellent performance, and contains features that provide flexibility and low encoding/decoding complexity.

- **Structured block LDPC for low complexity decoding**. The entire matrix (i.e., both the sections that correspond to the information and the parity) is composed of the same style of blocks, which reduces decoder implementation complexity and allows structured decoding.
- **Low-complexity differential-style encoding**. The encoding can be performed in a structured, recursive manner, without hurting performance with multiple weight-1 columns.
- **Compact representation**. The shift values for each block size are derived from the largest block size of that code rate, facilitating the representation and implementation of the encoder/decoder.
- **Semi-regular R=3/4 code**. The R=3/4 code offers the potential complexity reduction of a semi-regular matrix (i.e., regular on the portion of the matrix corresponding to the information bits) with performance very close to an irregular code.
- **Simplified structured decoder architecture.** Each base matrix has 24 columns, simplifying the implementation. Having the same number of columns between code rates minimizes the number of different expansion factors that must be supported. Having 24 columns provides better performance than fewer than 24 columns, and provides a larger minimum parallelization than a design with more than 24 columns.
- **Enhanced Layered Decoding**. The R=1/2 code is designed such that a row permutation of [0, 2, 4, 11, 6, 8, 10, 1, 3, 5, 7, 9] may be applied to the model matrix prior to layered decoding. After permutation, consecutive rows do not intersect within an iteration as well as between two iterations. For some layered decoder architectures, this feature can be used to effectively double R=1/2 throughput through pipelining.

LDPC codes offer similar or better performance than turbo codes, with the potential for lower decoder complexity. The defined LDPC code is designed to match the 802.16 OFDMA subchannel structure, and does not require puncturing or rate-matching operations to provide each code rate / block size.

For convenience, some of the specific code properties are summarized below.

Code rate	**1/2**	**2/3**	**3/4**
Model matrix size	12×24	8×24	6×24
Information portion	Irregular	Irregular	Regular
Maximum column weight	6	6	4
Method of modifying the shift sizes	Scale+floor	Modulo	Scale+floor

Simulation Results

Simulation results for rate 1/2, 2/3, and 3/4 code families are shown for AWGN and QPSK in Figures 1, 2, and 3, respectively. For each code rate, 19 *z* values ranging from 24 to 96 are shown, which correspond to 19 code lengths with *n* ranging from 576 to 2304. The simulations used a maximum of 50 iterations and generic floating-point belief propagation. The design targets were to have no floor down to 10^{-4} FER, and for the code performance to improve

with larger code lengths. Fading channel results for the ITU-B channel and QPSK are shown in Figures 4, 5, and 6. Significant gains are provided over the convolutional code.

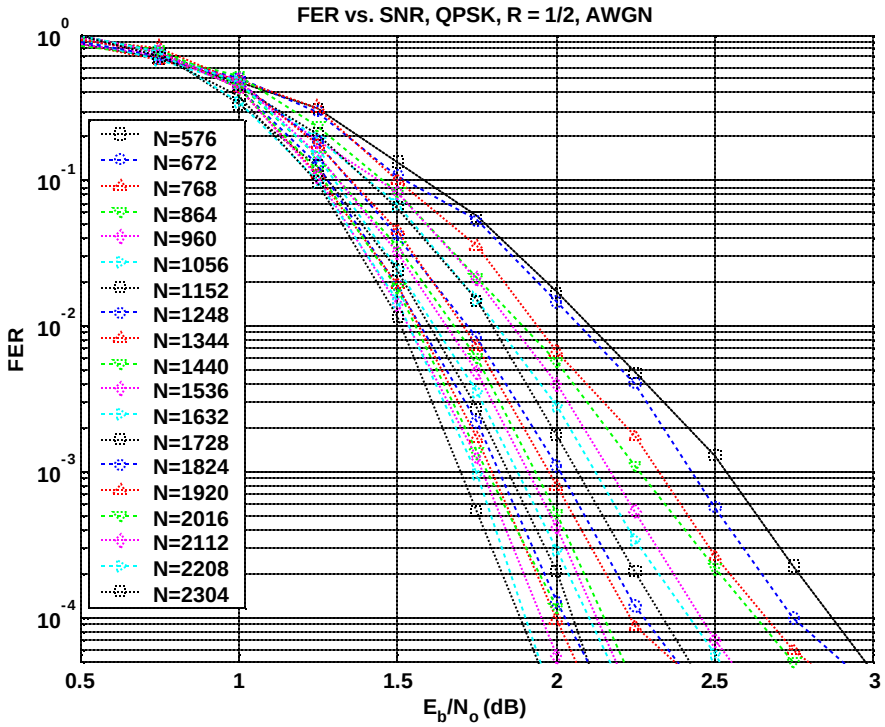

Figure 1. FER vs. E_b/N_0 (dB), QPSK, rate 1/2, AWGN channel.

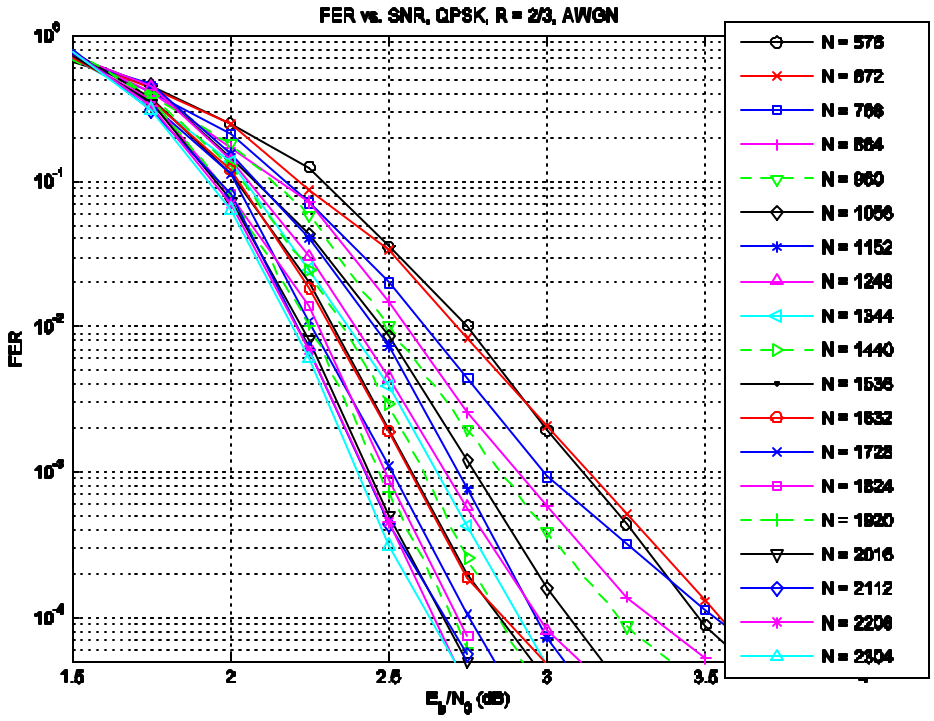

Figure 2. FER vs. E_b/N_0 (dB), QPSK, rate 2/3, AWGN channel.

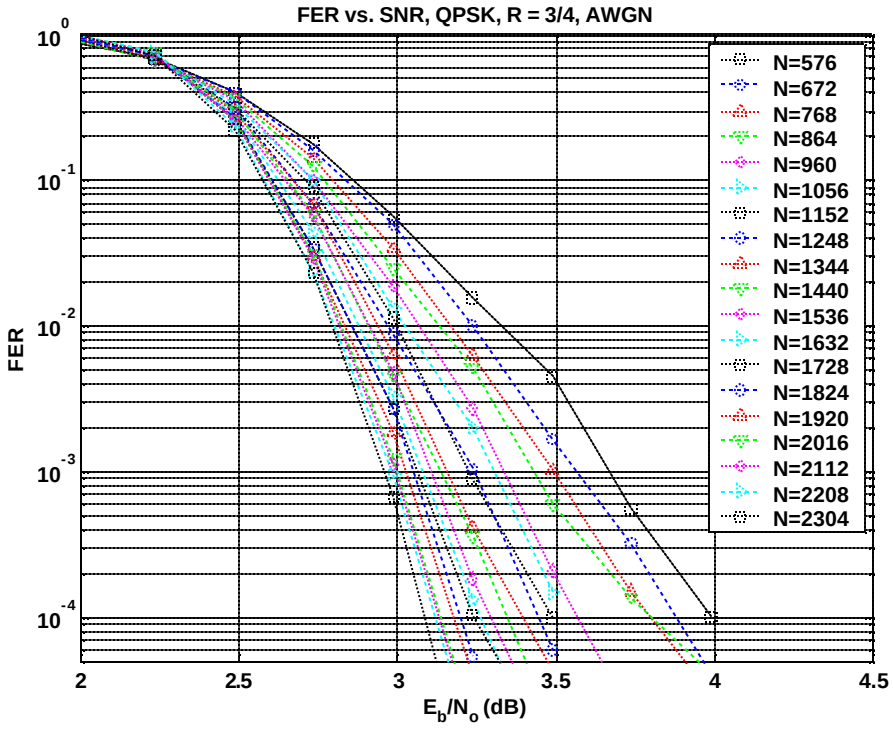

Figure 3. FER vs. E_b/N_0 (dB), QPSK, rate 3/4, AWGN channel.

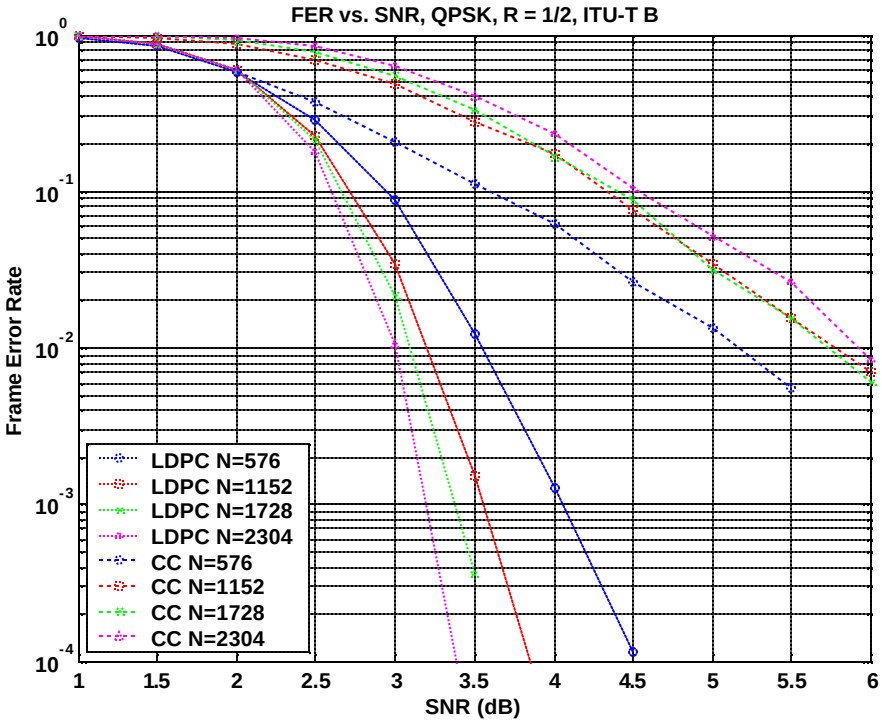

Figure 4. FER vs. SNR (dB), QPSK, rate 1/2, ITU-B channel.

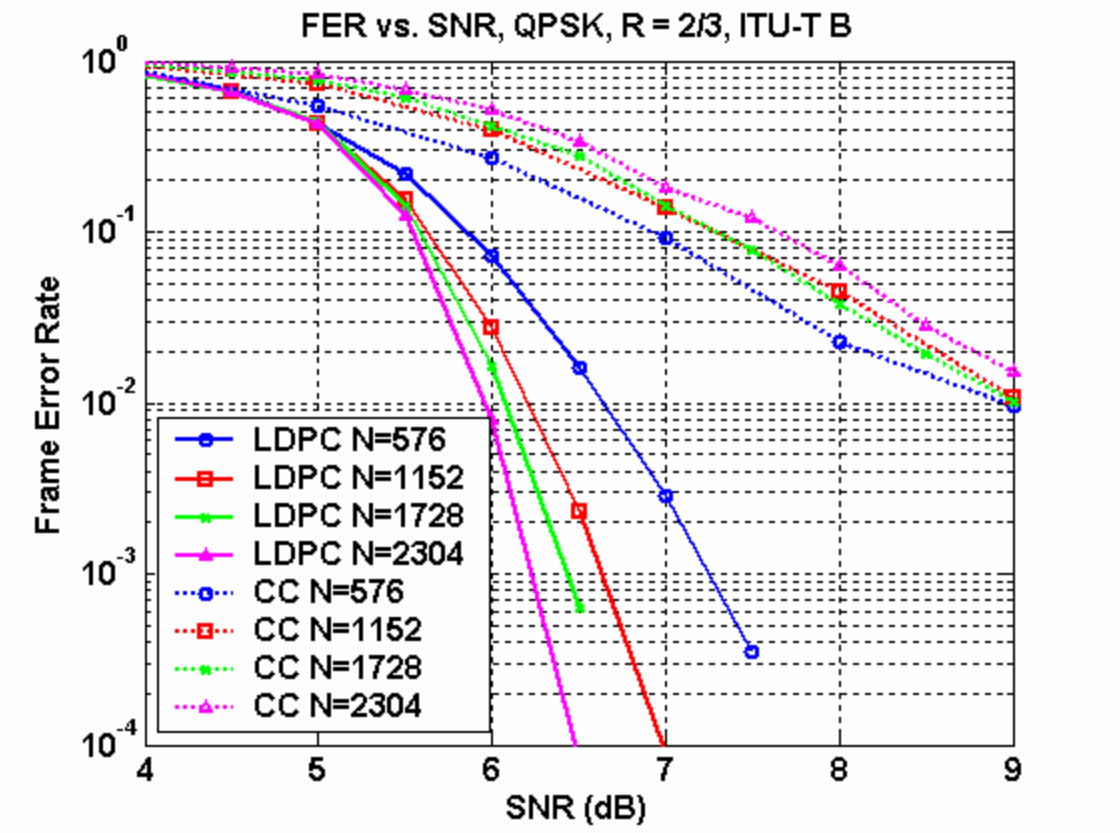

Figure 5. FER vs. SNR (dB), QPSK, rate 2/3, ITU-B channel.

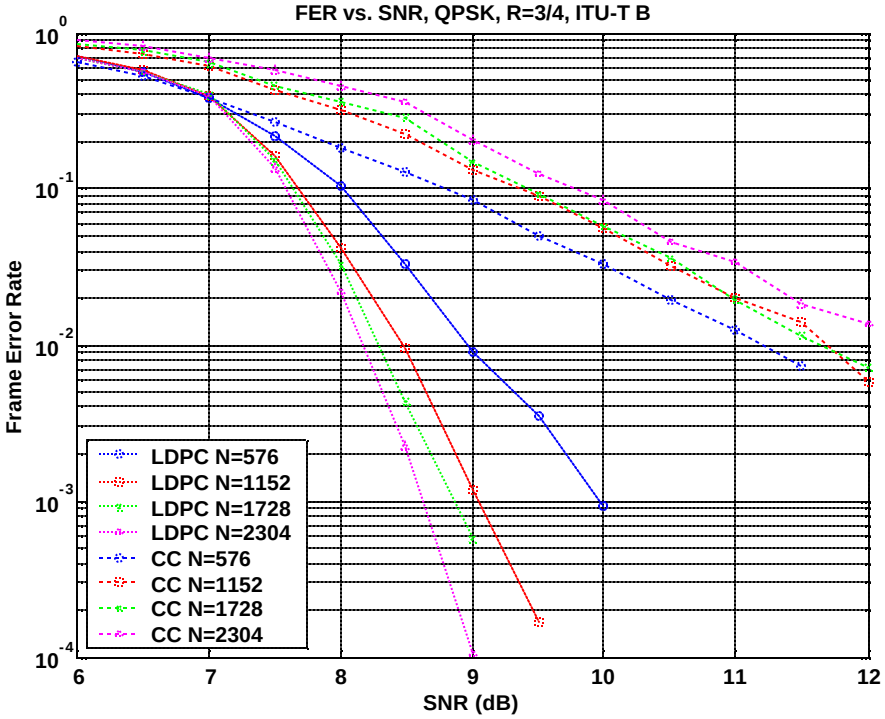

Figure 6. FER vs. SNR (dB), QPSK, rate 3/4, ITU-B channel.

Parallel Encoding Method:

Parallel encoding method is similar to Direct Encoding Method 1 but almost all parity check parity bits are concurrently generated as follows:

1) Initialization. The parity check bit vector $v(0)$ is computed by

$$v(0) = \sum_{j=0}^{k_b-1}\left(\sum_{q=0}^{m_b-1} P_{p(q,j)}\right)u(j) \quad (1)$$

2) Parallel computation. The parity check bit vectors $v(1) \sim v(m_b - 1)$ are concurrently computed by

$$v(i) = \sum_{j=0}^{k_b-1}\left(\sum_{q=i}^{m_b-1} P_{p(q,j)}\right)u(j) + \sum_{q=i}^{m_b-1} P_{p(q,k_b)}v(0) \quad i = 1,\cdots,m_b - 1 \quad (2)$$

After the parity-check bit vector $v(0)$ is initialized, all remainder parity check bit vectors could be computed simultaneously according to (2). Thus the encoding latency can significantly reduced at the expense of extra storage. Expression $\left(\sum_{q=i}^{m_b-1} P_{p(q,j)}\right)$ in (1)~(2) are the summations of the specified rows in the model matrix, and the summations can be stored beforehand.

Recommended Text Changes:

Modify the text in 802.16e_D5a as follows, adjusting the numbering as required:

<Delete the text "of scaling and shortening" in section 8.4.9.2.5.1, p. 364, line 22>

<Move the text between the section headings "Direct Encoding (Informative)" and "Method 1" (section 8.4.9.2.5.2 p. 365 line 35 to p366 line 9) to section 8.4.9.2.5.1 Code Description p. 364 line 50, immediately after the sentence "The base matrix $\mathbf{H}_b$ is partitioned into two sections …" Delete "For the two methods, described below" from the moved text.>

<Add the following text to the end of section 8.4.9.2.5.2 "Direct Encoding (Informative)" "Method 1", p. 367 line 59>
Equivalently, Method 1 can be implemented in a parallel fashion where almost all parity check parity bits are generated simultaneously. The initialization and the recursion steps of Method 1 become

1) Initialization. The parity check bit vector $v(0)$ are computed by

$$v(0) = \sum_{j=0}^{k_b-1}\left(\sum_{q=0}^{m_b-1} P_{p(q,j)}\right)u(j) \qquad (1)$$

2) Parallel computation. The parity check bit vectors $v(1) \sim v(m_b - 1)$ are concurrently computed by

$$v(i) = \sum_{j=0}^{k_b-1}\left(\sum_{q=i}^{m_b-1} P_{p(q,j)}\right)u(j) + \sum_{q=i}^{m_b-1} P_{p(q,k_b)}v(0) \quad i = 1,\cdots,m_b - 1 \qquad (2)$$

The parallel encoding method may significantly reduced the latency at the expense of extra storage for the sum $\left(\sum_{q=i}^{m_b-1} P_{p(q,j)}\right)$.

<Add the following text to the end of section 8.4.9.2.5.1 Code Description.>
A base model matrix is defined for the largest code length (n=2304) of each code rate. The set of shifts $\{p(i,j)\}$ in the base model matrix are used to determine the shift sizes for all other code lengths of the same code rate. Each base model matrix has n_b=24 columns, and the expansion factor z_f is equal to $n/24$ for code length n. For code length n=2304 the expansion factor is designated z_0=96.
For code rates 1/2, ~~and~~ 3/4 A and B code, and 2/3 B code, the shift sizes $\{p(f, i, j)\}$ for a code size corresponding to expansion factor z_f are derived from $\{p(i,j)\}$ by scaling $p(i,j)$ proportionally,

$$p(f,i,j) = \begin{cases} p(i,j), & p(i,j) \le 0 \\ \left\lfloor \dfrac{p(i,j)z_f}{z_0} \right\rfloor, & p(i,j) > 0 \end{cases},$$

where ëxû denotes the flooring function which gives the nearest integer towards -∞.

For code rate 2/3 A code, the shift sizes $\{p(f, i, j)\}$ for a code size corresponding to expansion factor z_f are derived from $\{p(i,j)\}$ by using a modulo function

$$p(f,i,j) = \begin{cases} p(i,j), & p(i,j) \le 0 \\ \mathrm{mod}\left(p(i,j), z_f\right), & p(i,j) > 0 \end{cases}.$$

Rate 1/2:

```
-1 94 73 -1 -1 -1 -1 -1 55 83 -1 -1  7  0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 27 -1 -1 -1 22 79  9 -1 -1 -1 12 -1  0  0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 24 22 81 -1 33 -1 -1 -1  0 -1 -1  0  0 -1 -1 -1 -1 -1 -1 -1 -1
61 -1 47 -1 -1 -1 -1 -1 65 25 -1 -1 -1 -1 -1  0  0 -1 -1 -1 -1 -1 -1 -1
-1 -1 39 -1 -1 -1 84 -1 -1 41 72 -1 -1 -1 -1 -1  0  0 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 46 40 -1 82 -1 -1 -1 79  0 -1 -1 -1 -1  0  0 -1 -1 -1 -1 -1
-1 -1 95 53 -1 -1 -1 -1 -1 14 18 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1 -1 -1
-1 11 73 -1 -1 -1  2 -1 -1 47 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1 -1
12 -1 -1 -1 83 24 -1 43 -1 -1 -1 51 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1
-1 -1 -1 -1 -1 94 -1 59 -1 -1 70 72 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1
-1 -1  7 65 -1 -1 -1 -1 39 49 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0
43 -1 -1 -1 -1 66 -1 41 -1 -1 -1 26  7 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0
```

Rate 2/3 A code:

```
 3  0 -1 -1  2  0 -1  3  7 -1  1  1 -1 -1 -1 -1  1  0 -1 -1 -1 -1 -1 -1
-1 -1  1 -1 36 -1 -1 34 10 -1 -1 18  2 -1  3  0 -1  0  0 -1 -1 -1 -1 -1
-1 -1 12  2 -1 15 -1 40 -1  3 -1 15 -1  2 13 -1 -1 -1  0  0 -1 -1 -1 -1
-1 -1 19 24 -1  3  0 -1  6 -1 17 -1 -1 -1  8 39 -1 -1 -1  0  0 -1 -1 -1
20 -1  6 -1 -1 10 29 -1 -1 28 -1 14 -1 38 -1 -1  0 -1 -1 -1  0  0 -1 -1
-1 -1 10 -1 28 20 -1 -1  8 -1 36 -1  9 -1 21 45 -1 -1 -1 -1 -1  0  0 -1
35 25 -1 37 -1 21 -1 -1  5 -1 -1  0 -1  4 20 -1 -1 -1 -1 -1 -1 -1  0  0
-1  6  6 -1 -1 -1  4 -1 14 30 -1  3 36 -1 14 -1  1 -1 -1 -1 -1 -1 -1  0
```

Rate 2/3 B code:

```
 2 -1 19 -1 47 -1 48 -1 36 -1 82 -1 47 -1 15 -1 95  0 -1 -1 -1 -1 -1 -1
-1 69 -1 88 -1 33 -1  3 -1 16 -1 37 -1 40 -1 48 -1  0  0 -1 -1 -1 -1 -1
10 -1 86 -1 62 -1 28 -1 85 -1 16 -1 34 -1 73 -1 -1 -1  0  0 -1 -1 -1 -1
-1 28 -1 32 -1 81 -1 27 -1 88 -1  5 -1 56 -1 37 -1 -1 -1  0  0 -1 -1 -1
23 -1 29 -1 15 -1 30 -1 66 -1 24 -1 50 -1 62 -1 -1 -1 -1 -1  0  0 -1 -1
-1 30 -1 65 -1 54 -1 14 -1  0 -1 30 -1 74 -1  0 -1 -1 -1 -1 -1  0  0 -1
32 -1  0 -1 15 -1 56 -1 85 -1  5 -1  6 -1 52 -1  0 -1 -1 -1 -1 -1  0  0
-1  0 -1 47 -1 13 -1 61 -1 84 -1 55 -1 78 -1 41 95 -1 -1 -1 -1 -1 -1  0
```

Rate 3/4 A code:

```
6  38  3 93 -1 -1 -1 30 70 -1 86 -1 37 38  4 11 -1 46 48  0 -1 -1 -1 -1
62 94 19 84 -1 92 78 -1 15 -1 -1 92 -1 45 24 32 30 -1 -1  0  0 -1 -1 -1
```

```
71 -1 55 -1 12 66 45 79 -1 78 -1 -1 10 -1 22 55 70 82 -1 -1  0  0 -1 -1
38 61 -1 66  9 73 47 64 -1 39 61 43 -1 -1 -1 -1 95 32  0 -1 -1  0  0 -1
-1 -1 -1 -1 32 52 55 80 95 22  6 51 24 90 44 20 -1 -1 -1 -1 -1 -1  0  0
-1 63 31 88 20 -1 -1 -1  6 40 56 16 71 53 -1 -1 27 26 48 -1 -1 -1 -1  0
```

Rate 3/4 B code:

```
-1 81 -1 28 -1 -1 14 25 17 -1 -1 85 29 52 78 95 22 92  0  0 -1 -1 -1 -1
42 -1 14 68 32 -1 -1 -1 -1 70 43 11 36 40 33 57 38 24 -1  0  0 -1 -1 -1
-1 -1 20 -1 -1 63 39 -1 70 67 -1 38  4 72 47 29 60 5  80 -1  0  0 -1 -1
64  2 -1 -1 63 -1 -1  3 51 -1 81 15 94  9 85 36 14 19 -1 -1 -1  0  0 -1
-1 53 60 80 -1 26 75 -1 -1 -1 -1 86 77  1  3 72 60 25 -1 -1 -1 -1  0  0
77 -1 -1 -1 15 28 -1 35 -1 72 30 68 85 84 26 64 11 89  0 -1 -1 -1 -1  0
```

<Replace the contents of section 8.4.9.2.5.3 (p. 369 line 45 to p. 370 line 64) with the following text and table.>
The LDPC code flexibly supports different block sizes for each code rate through the use of an expansion factor. Each base model matrix has n_b=24 columns, and the expansion factor (z factor) is equal to n/24 for code length n. In each case, the number of information bits is equal to the code rate times the coded length n.

Table 316b – LDPC Block Sizes and Code Rates

n (bits)	*n* (bytes)	*z* factor	*k* (bytes)			Number of subchannels		
			R=1/2	R=2/3	R=3/4	QPSK	16QAM	64QAM
576	72	24	36	48	54	6	3	2
672	84	28	42	56	63	7		
768	96	32	48	64	72	8	4	
864	108	36	54	72	81	9		3
960	120	40	60	80	90	10	5	
1056	132	44	66	88	99	11		
1152	144	48	72	96	108	12	6	4
1248	156	52	78	104	117	13		
1344	168	56	84	112	126	14	7	
1440	180	60	90	120	135	15		5
1536	192	64	96	128	144	16	8	
1632	204	68	102	136	153	17		
1728	216	72	108	144	162	18	9	6
1824	228	76	114	152	171	19		

1920	240	80	120	160	180	20	10	
2016	252	84	126	168	189	21		7
2112	264	88	132	176	198	22	11	
2208	276	92	138	184	207	23		
2304	288	96	144	192	216	24	12	8

<In 8.4.9.2.5.4 Packet Encoding, p.371 lines 19-23, correct the formatting by inserting delimiters ": " after the variables "j"(line 19), "Nsch" (line 20), "F" (line 21), and "M" (line 22).>

<In 8.4.9.2.5.4 Packet Encoding, p.371 line 19, delete the text "and FEC rate" because the parameter j is independent of the FEC rate for LDPC>

<NOTE to Editor: The remaining items below update the signaling for LDPC>

<In 8.4.4.3 DL Frame Prefix p. 232, line 48, Table 266a, "Coding_Indication" row, insert a line in the "Notes" column "0b100 – LDPC encoding used on DL-MAP", and modify existing line "0b100 to 0b111 – Reserved" to "0b101 to 0b111 – Reserved">

<In 11.3.1, p. 390, line 57 add the following text>
11.3.1.1 Uplink burst profile encodings
[Insert the following text in the "Value" column of the first row ("FEC code type and modulation type") of Table 355 p. 663 of 802.16-REVd/D5, and change "26..255=Reserved" to "~~35~~41..255=Reserved"]
26=QPSK (LDPC) 1/2
27=QPSK (LDPC) 2/3 A code
28=QPSK (LDPC) 3/4 A code
29=16-QAM (LDPC) 1/2
30=16-QAM (LDPC) 2/3 A code
31=16-QAM (LDPC) 3/4 A code
32=64-QAM (LDPC) 1/2
33=64-QAM (LDPC) 2/3 A code
34=64-QAM (LDPC) 3/4 A code
35=QPSK (LDPC) 2/3 B code
36=QPSK (LDPC) 3/4 B code
37=16-QAM (LDPC) 2/3 B code
38=16-QAM (LDPC) 3/4 B code
39=64-QAM (LDPC) 2/3 B code

40=64-QAM (LDPC) 3/4 B code

<In 11.4, p. 394, line 56 add the following text>

11.4.2 Downlink burst profile encodings

[Insert the following text in the "Value" column of the first row ("FEC code type") of Table 361 p. 668 of 802.16-REVd/D5, and change "26..255=Reserved" to "~~35~~41..255=Reserved"]

26=QPSK (LDPC) 1/2
27=QPSK (LDPC) 2/3 A code
28=QPSK (LDPC) 3/4 A code
29=16-QAM (LDPC) 1/2
30=16-QAM (LDPC) 2/3 A code
31=16-QAM (LDPC) 3/4 A code
32=64-QAM (LDPC) 1/2
33=64-QAM (LDPC) 2/3 A code
34=64-QAM (LDPC) 3/4 A code
35=QPSK (LDPC) 2/3 B code
36=QPSK (LDPC) 3/4 B code
37=16-QAM (LDPC) 2/3 B code
38=16-QAM (LDPC) 3/4 B code
39=64-QAM (LDPC) 2/3 B code
40=64-QAM (LDPC) 3/4 B code

<In 11.8.3.7.2 OFDMA MSS demodulator, p. 408, line 24 change the "Length" column entry of "1" to "2", and insert the following two entries after line 31 in the "Value" column>

Bit#8: LDPC
Bits#9-15: Reserved; shall be set to zero

<In 11.8.3.7.2 OFDMA MSS demodulator, p. 408, line 32 add the following text>

11.8.3.7.3 OFDMA MSS modulator

[Copy the table from 11.8.3.7.3, and insert the following text in the "Value" column of the first row, and change "Bits#6-7: Reserved; shall be set to zero" to "Bit#7: Reserved; shall be set to zero"]

Bit#6: LDPC